Come inviare email con Python

Inviare email con Python è un compito comune, soprattutto per applicazioni che richiedono notifiche, report automatici o conferme. Grazie al modulo `smtplib` integrato e a librerie aggiuntive come `email`, è possibile automatizzare questo processo facilmente.

Requisiti

- Python installato sul tuo sistema
- Accesso a un server SMTP (come Gmail, Outlook, o il tuo server aziendale)

Codice di Base

Ecco un esempio di base per inviare un'email usando Gmail come server SMTP:

```
import smtplib
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart

def send_email():
    # Configurazione email
    sender = "tuoindirizzo@server.com"
    recipient = "destinatario@example.com"
    password = "tuapassword"

    # Creazione del messaggio
    subject = "Oggetto dell'Email"
    body = "Questo è il contenuto dell'email."

    msg = MIMEMultipart()
    msg["From"] = sender
    msg["To"] = recipient
    msg["Subject"] = subject
    msg.attach(MIMEText(body, "plain"))

    try:
        # Connessione al server SMTP
        with smtplib.SMTP("smtp.server.com", 587) as server:
            server.starttls()
            server.login(sender, password)
            server.sendmail(sender, recipient, msg.as_string())
        print("Email inviata con successo!")
    except Exception as e:
        print(f"Errore nell'invio dell'email: {e}")
```

```
# Chiamata alla funzione
send_email()
```

Dettagli e Sicurezza

Per proteggere la tua password, considera l'uso di variabili d'ambiente o un file di configurazione separato. Se utilizzi Gmail, potresti dover abilitare l'accesso di app meno sicure o configurare una password per applicazioni specifiche.

Estensioni

Per inviare email più complesse, come quelle con allegati o formattazione HTML, puoi estendere il codice aggiungendo oggetti MIME appropriati.

```
from email.mime.base import MIMEBase
from email import encoders

# Aggiunta di un allegato
filename = "document.pdf"
with open(filename, "rb") as attachment:
    part = MIMEBase("application", "octet-stream")
    part.set_payload(attachment.read())

encoders.encode_base64(part)
part.add_header(
    "Content-Disposition",
    f"attachment; filename={filename}",
)
msg.attach(part)
```

Conclusione

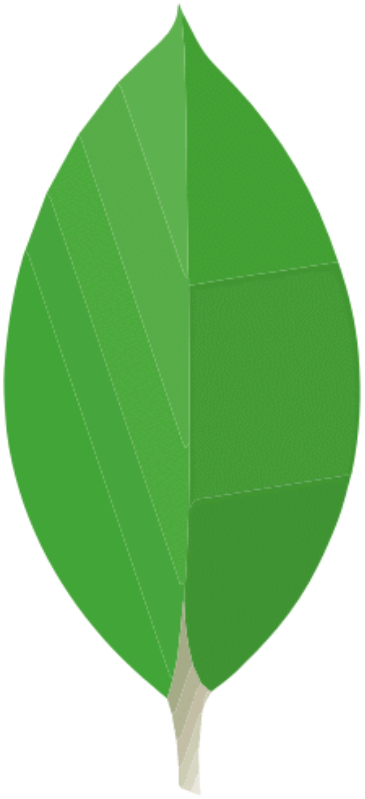
Con queste basi, sei pronto per integrare la funzionalità di invio email nelle tue applicazioni Python.

Applicazioni Correlate

-

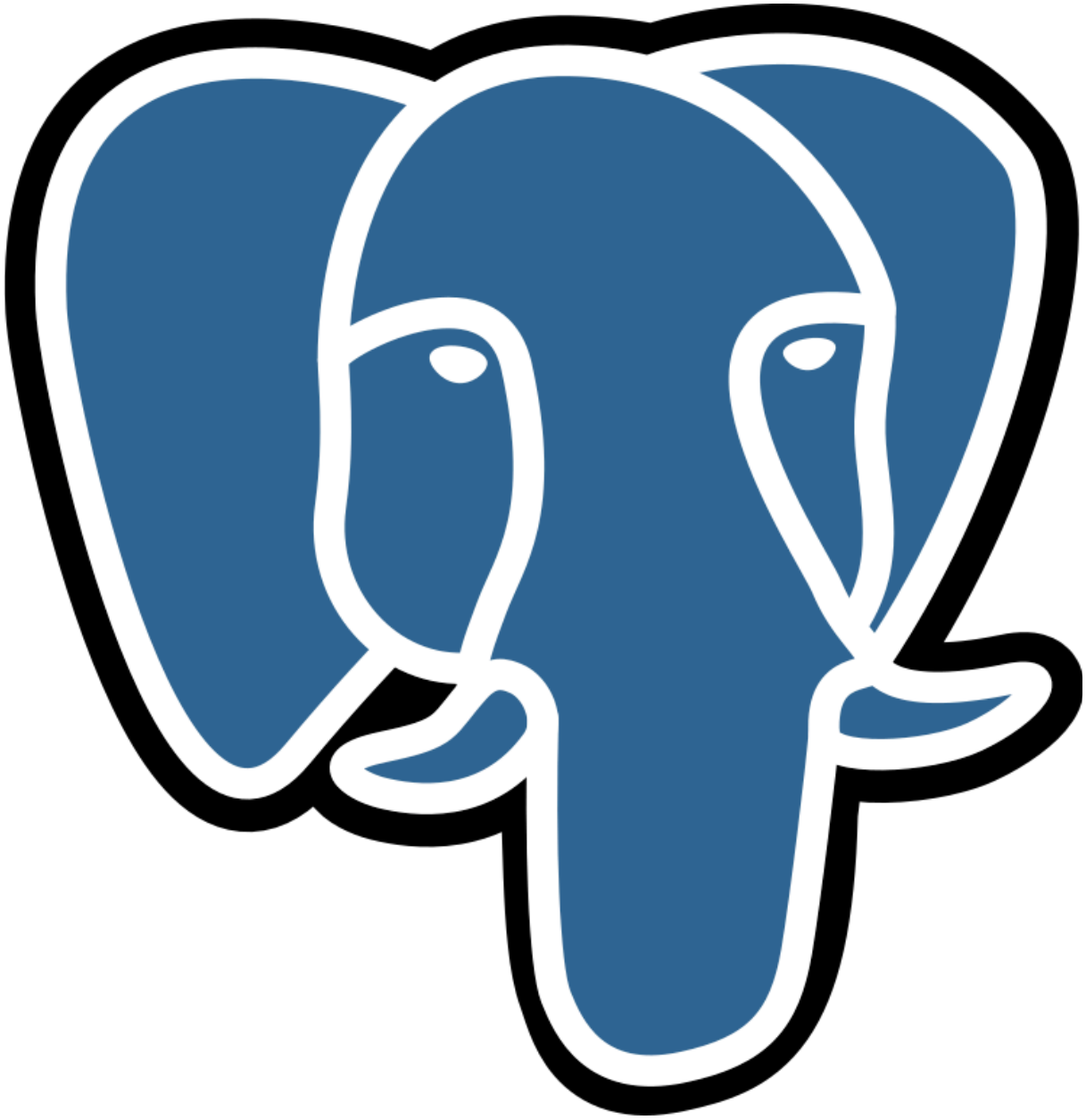
Python Placeholder Image

Applicazione sviluppata in Python con Flask per la creazione di immagini segnaposto.
DockerDocker ComposePythonFlaskJavaScript



Python MongoDB App

Applicazione basata su MongoDB e sviluppata in Python e Flask.
DockerDocker ComposePythonFlaskMongoDB



Python PostgreSQL App

Applicazione basata su PostgreSQL con Python e Flask.
DockerDocker ComposePythonFlask